ГЛАВА XX. Функции ошибки в задачах регрессии

Если не можешь что-то измерить, то не сможешь это улучшить.
У. Томсон

Умный человек не делает сам все ошибки – он дает шанс и другим.
У. Черчилль

Построить какой-нибудь алгоритм машинного обучения не так сложно, например, при прогнозе спроса на товары можно считать, что на следующей неделе каждого вида товара будет куплено ровно столько, сколько и на этой. После того, как алгоритм построен, возникает естественный вопрос: можно ли его улучшить? Для этого надо уметь измерять качество (performance) работы алгоритма, чтобы было понятно, что улучшать.

Простая эвристика «завтра будет так же, как вчера» очень эффективна. Продвинутые методы машинного обучения часто улучшают её лишь ненамного.

В этом разделе считаем, что есть некоторая выборка (на которой мы хотим измерить качество алгоритма) $x_1,\ldots,x_m$ с известными целевыми значениями (метками)

$$y_1 = y(x_1),\ldots,y_m = y(x_m).$$

В данном случае не важно, обучающая это выборка или контрольная – мы на ней будем измерять качество ответов алгоритма $a(x)$. Как отмечается в главе «Контроль», качество или ошибку лучше оценивать на контрольной выборке, но чтобы их оптимизировать на обучающей, их всё равно надо уметь вычислять. Хотя не всегда при обучении алгоритмов машинного обучения ошибка минимизируется напрямую, часто используется суррогатные функции ошибки[1], см. главу «Линейные классификаторы». Считаем, что на этой выборке алгоритм выдал ответы $a_1 = a(x_1),\ldots,a_m = a(x_m)$.

[1] В последние годы функцию ошибки, которую уменьшают при обучении модели, называют «функцией потери» (loss function). Хотя изначально «потерями» назывались ошибки в байесовском подходе.

Ясно, что надо придумать функцию, которая измеряет схожесть полученных ответов и истинных. Есть два вида таких функций:

- **функционалы качества** (измеряют описанную похожесть),
- **функции ошибки** (измеряют различие между правильным ответом и полученным).

Из семантики понятно, что функционал качества максимизируют при обучении (настройке параметров алгоритма), а функцию ошибки, наоборот, минимизируют. В английском варианте есть общее понятие **метрики качества (metrics)**, которое объединяет перечисленные выше понятия. В русском языке термин «метрика» и так очень нагружен, но тоже применяется в смысле оценки качества, однако лучше использовать термин **показатель качества**.

Не плодите метрик без необходимости;)

Ниже перечислим стандартные показатели качества, будем стараться указывать оптимальный алгоритм в классе константных, поскольку

- его часто можно найти аналитически,
- с помощью него иллюстрируются важные особенности функции ошибки,
- хорошее решение может строиться из таких константных решений (например, случайный лес – это сумма решающих деревьев, каждое из которых является кусочно-константной функцией).

Отметим ещё одну особенность употребления терминов. Под качеством алгоритма нам бы хотелось понимать качество его работы (в среднем или в худшем случае) при его эксплуатации. Поскольку мы часто лишены возможности быстро и эффективно оценить работу алгоритма на данных, которых, быть может, ещё и нет – они появятся в будущем, под качеством понимают **качество ответа** на конкретной выборке. Именно поэтому все рассматриваемые функционалы качества и функции ошибки имеют вид:

$$L((y_1,\ldots,y_m),(a_1,\ldots,a_m)).$$

Очень часто, но далеко не всегда, функция ошибки на выборке представляется в виде сумм ошибок на отдельных объектах:

$$\frac{1}{m}\sum_{i=1}^{m} L(y_i,a_i)$$

(здесь для обозначения ошибки на выборке и на отдельном объекте мы используем одно и то же обозначение L, что не должно вызвать путаницы).

В этой главе опишем функции ошибки для задачи регрессии, которая символически изображена на рис XX.1: точки – элементы выборки, кривая – ответы алгоритма регрессии, отрезки – ошибки на элементах выборки. Понятно, что наиболее естественно при вычислении ошибки на всей выборке учитывать ошибки $(a_i - y_i)$ на каждом объекте.

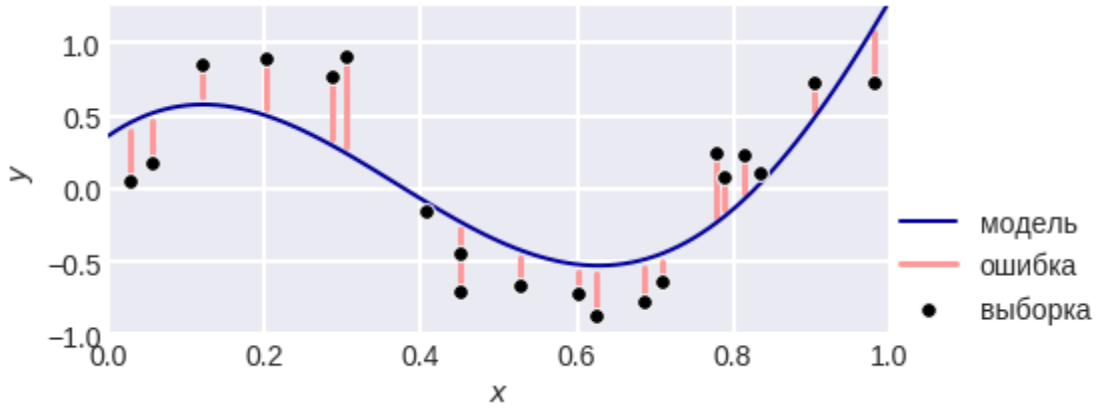

Рис. XX.1. Иллюстрация задачи регрессии.

Средний модуль отклонения (**MAE – Mean Absolute Error** или **MAD – Mean Absolute Deviation**):

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m} | a_i - y_i | .$$

Полезно помнить (см. главу «Средние»), что оптимальный константный алгоритм с точки зрения этой функции ошибки:

$$a = \text{median}(\{y_i\}_{i=1}^{m}),$$

т.е. решением задачи

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m} | a - y_i | \to \min_a \qquad \text{(XX.01)}$$

в классе констант является медиана, см. рис. XX.2 – показаны графики слагаемых в (XX.01) и график MAE от значения a, минимум достигается в средней точке (если бы точек было нечётное число, то на отрезке между двумя средними).

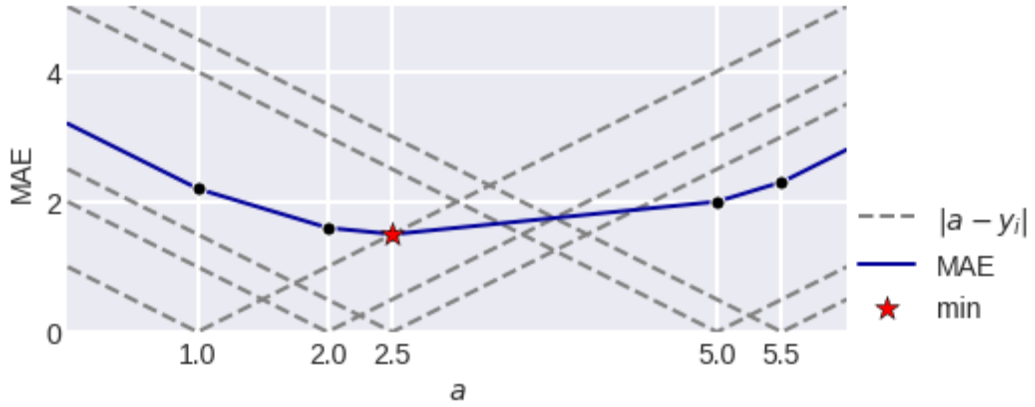

Рис. XX.2. График MAE от константного ответа a в модельной задаче.

В задачах регрессии с MAE при ансамблировании часто вместо усреднения нескольких алгоритмов берут их медиану – это, как правило, повышает качество. Если в такой задаче целевой признак целочисленный, то округление ответа часто не ухудшает качество решения.

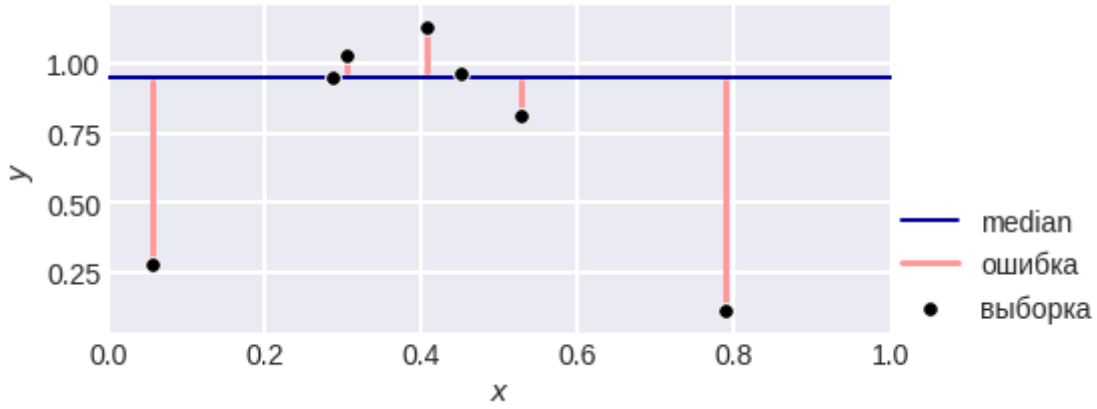

Рис. XX.3. Решение в классе константных алгоритмов.

Приведём теоретическое обоснование разумности использования MAE как функции ошибки. Пусть целевой признак описывается моделью

$$y = a_w(x) + \varepsilon$$

с точностью до шума $\varepsilon \sim \text{laplace}(0, \alpha)$, которое распределено по Лапласу, см. рис. XX.4. Для оценки параметров w модели $a_w(x)$ выписываем правдоподобие модели

$$p(y \mid x, w) = \frac{\alpha}{2} \exp\left[-\alpha \mid y - a_w(x) \mid\right].$$

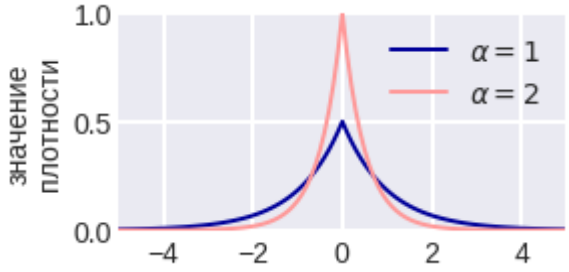

Рис. XX.4. Плотность распределения Лапласа.

Применение метода максимального правдоподобия (ММП) соответствует максимизации (делается предположение о независимости и одинаковом распределении ошибок):

$$\log L(w) = \log \prod_{i=1}^{m} p(y_i \mid x_i, w) =$$

$$= \sum_{i=1}^{m} \left[\log \frac{\alpha}{2} - \alpha \mid y_i - a_w(x_i) \mid \right] \to \max$$

или (отбрасывая не влияющую на решение константу)

$$\alpha \sum_{i=1}^{m} \mid y_i - a_w(x_i) \mid \to \min .$$

Таким образом, максимизация правдоподобия эквивалентна минимизации MAE. Заметим, что если бы в каждой точке была бы своя ошибка:

$$y_i = a_w(x_i) + \varepsilon_i,\ \varepsilon_i \sim \text{laplace}(0, \alpha_i),$$

то мы бы получили **взвешенный вариант MAE (weighted MAE):**

$$\sum_{i=1}^{m} \alpha_i \mid y_i - a_w(x_i) \mid \to \min ,$$

проведите соответствующие выкладки. Отметим, что **мы не делали никаких предположений о природе модели, только о распределении ошибок.**

Вот почему рекомендуют строить т.н. «Residual Plots» – важно знать распределение ошибок!

Средний квадрат отклонения (**MSE – Mean Squared Error**):

$$\text{MSE} = \frac{1}{m} \sum_{i=1}^{m} \mid a_i - y_i \mid^2$$

или корень из этой ошибки: **RMSE – Root Mean Squared Error** или **RMSD – Root Mean Square Deviation**

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{m}\sum_{i=1}^{m} | a_i - y_i |^2} \ .$$

Также часто используют т.н. **коэффициент детерминации (R^2):**

$$\text{R}^2 = 1 - \frac{\sum_{i=1}^{m} | a_i - y_i |^2}{\sum_{i=1}^{m} | \overline{y} - y_i |^2}, \ \overline{y} = \frac{1}{m}\sum_{i=1}^{m} y_i$$

Понятно, что с точки зрения оптимизации эти функции эквивалентны. MSE совсем простая функция, но её значения сложно интерпретировать, поэтому в RMSE значение приводится в масштаб изменений целевого вектора, но сложности интерпретации остаются. Оптимальное решение в задаче минимизации MSE и RMSE в классе констант – среднее арифметическое

Хорошо или плохо, если RMSE=27?

$$\overline{y} = \frac{1}{m}\sum_{i=1}^{m} y_i = \arg\min_a \frac{1}{m}\sum_{i=1}^{m} | a - y_i |^2 ,$$

(легко доказать, например, приравняв производную к нулю, см. рис. XX.5-6), поэтому в коэффициенте детерминации происходит нормировка: MSE-ошибка алгоритма делится на MSE-ошибку оптимального константного алгоритма. Коэффициент детерминации, который по смыслу является функционалом качества, принимает максимальное значение 1 в случае абсолютно точного ответа, значение 0 – для оптимального константного ответа, но может быть и меньше нуля, если решение хуже такого константного. Кстати, коэффициент детерминации широко используется в статистике, и в общем случае определяется так:

$$R^2 = 1 - \frac{\mathbf{D}(y \mid x)}{\mathbf{D}(y)},$$

т.е. единица минус доля дисперсии случайной ошибки модели (или условной по факторам дисперсии зависимой переменной) в дисперсии зависимой переменной.

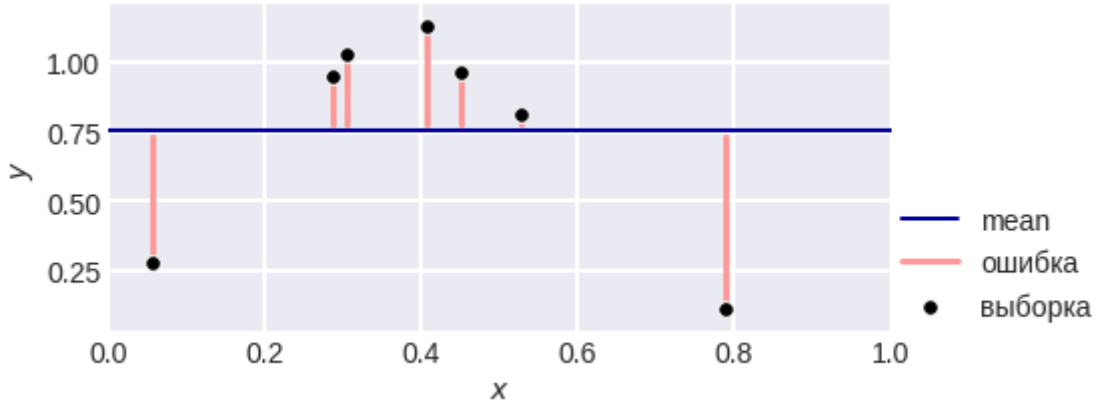

Рис. XX.5. Решение в классе константных алгоритмов.

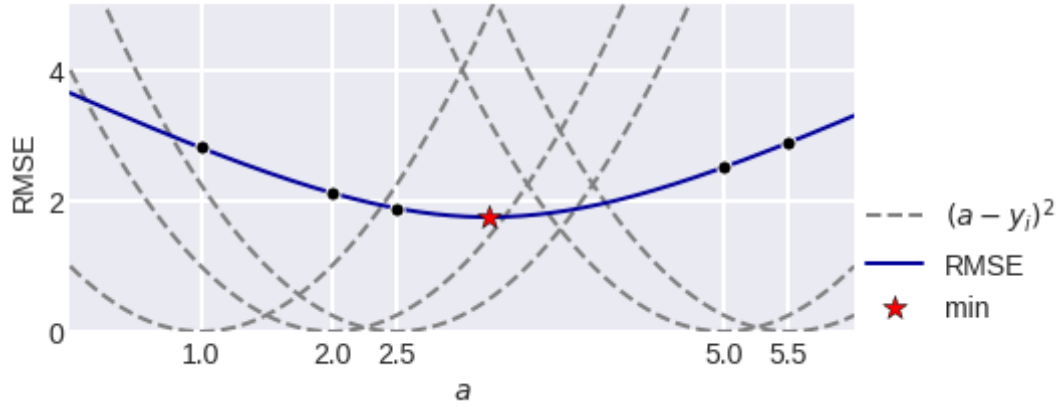

Рис. XX.6. График RMSE от константного ответа a в модельной задаче.

Заметим, что не смотря на то, что минимумы MSE, RMSE и максимум R^2 достигаются в одной точке, с точки зрения практической оптимизации, например методом градиентного спуска, эти функции не эквивалентны, поскольку у них разные производные, например

Хороший вопрос: какую функцию «выгоднее» оптимизировать на практике?

$$\frac{\partial \text{MSE}}{\partial a} = \frac{2}{m}\sum_{i=1}^{m}(a-y_i),\ \frac{\partial \text{RMSE}}{\partial a} = \frac{1}{m\,\text{RMSE}}\sum_{i=1}^{m}(a-y_i),$$

т.е. различаются в 2/RMSE раз. Отметим также, что модуль производной MAE не стремится к нулю при приближении к точке минимума MAE (в отличие от MSE).

Мы уже знаем, что среднее арифметическое не устойчиво к выбросам, в отличие от медианы. Это же справедливо в общем случае: если мы настраиваем параметры нашего алгоритма минимизируя (R)MSE, то они существенно зависят от выбросов. При использовании MAE зависимость, как правило, меньше. На рис. XX.7 показано, как будет выглядеть линейная модель, если её настраивать на (R)MSE / MAE на «хорошей» выборке (чёрные точки, модель

показана пунктиром) и выборке с выбросами (красные точки, модель показана сплошной линией). При добавлении выбросов в выборку коэффициенты прямой сильнее изменятся, если минимизировать (R)MSE.

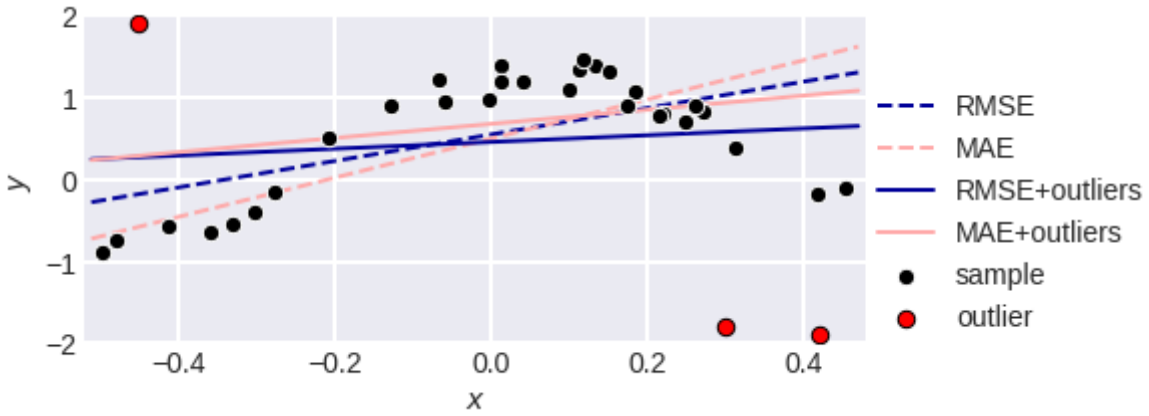

Рис. XX.7. Линейная регрессия в зависимости от минимизируемой ошибки и наличия выбросов.

Большинство алгоритмов регрессии явно или неявно ориентированы именно на минимизацию (R)MSE, например, линейная (гребневая) регрессия или случайный лес. При ансамблировании в задачах регрессии с (R)MSE лучше использовать «обычное» усреднение базовых алгоритмов (с помощью среднего арифметического). Отметим также, что в отличие от MSE и RMSE, R2 несимметричная функция:

$$\mathrm{R}^2((y_i)_{i=1}^m,(a_i)_{i=1}^m) \neq \mathrm{R}^2((a_i)_{i=1}^m,(y_i)_{i=1}^m).$$

Теоретическое обоснование (R)MSE подобно MAE, только теперь полагаем, что

$$y = a_w(x) + \varepsilon$$

с точностью до нормально распределённого шума $\varepsilon \sim \mathrm{norm}(0,\sigma^2)$, рис. XX.8. Для оценки параметров w модели $a_w(x)$ выписываем правдоподобие модели

$$p(y \mid x, w) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(y - a_w(x))^2}{2\sigma^2}\right]$$

и применяем метод максимального правдоподобия:

$$\log L(w) = \log \prod_{i=1}^{m} p(y_i \mid x_i, w) =$$

$$= \sum_{i=1}^{m} \left[-\frac{1}{2}\log(2\pi\sigma^2) - \frac{(y_i - a_w(x_i))^2}{2\sigma^2} \right] \to \max .$$

Откуда получаем задачу минимизации MSE:

$$\frac{1}{2\sigma^2} \sum_{i=1}^{m} (y_i - a_w(x_i))^2 \to \min$$

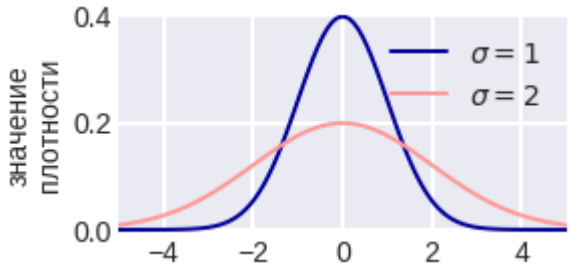

Рис. XX.8. Плотность нормального распределения.

Аналогично взвешенному MAE здесь можно вывести **взвешенную среднеквадратичную ошибку (weighted MSE)**. Обратите внимание, что вес у объекта обратно пропорционален дисперсии ошибки на нём. Таким образом, чем больше мы уверены в правильности целевого значения на объекте, тем с большим весом он входит в суммарный эмпирический риск:

$$\sum_{i=1}^{m} \frac{1}{\sigma_i^2} (y_i - a_w(x_i))^2 .$$

Есть ещё одно интересное (и не вероятностное!) обоснование среднеквадратичной ошибки[1]. Пусть ошибка на одном объекте представима в виде $l(y,a) = g(y-a)$, что вполне логично: зависит от отклонения истинного значения от нашего ответа. Также логично предположить:

1) $g(0) = 0$ – если ответ верный, то ошибка нулевая,

2) $|z_1| \le |z_2| \Rightarrow g(z_1) \le g(z_2)$ – чем больше отклонение, тем больше ошибка, можно также сразу предположить, что функция g чётная,

3) функция g достаточно гладкая и представима в виде ряда Маклорена:

[1] Преобразовано рассуждение из книги Лагутин М. Б. Наглядная математическая статистика. Учебное пособие. – БИНОМ. Лаборатория знаний, 2012.

$$g(z) = g(0) + g'(0)z + \frac{g''(0)}{2}z^2 + o(z^2),$$

но тогда, пренебрегая последним остаточным членом в этом ряде

$$l(y,a) = g(y-a) \approx g(0) + g'(0)(y-a) + \frac{g''(0)}{2}(y-a)^2$$

Предположение (1) обнуляет первое слагаемое $g(0)$, а из предположения (2) следует, что $g'(0) = 0$ и обнуляется второе слагаемое, таким образом

$$l(y,a) \approx C(y-a)^2.$$

Таким образом, с точностью до константы $C > 0$ (нулевая константа приводит к нулевой функции, а отрицательная противоречит предположению (2)), когда значения y и a близки (формула Маклорена верна ведь в малой окрестности нуля и мы пренебрегли остаточным членом), разумно использовать средний квадрат отклонения. Именно это делается в **функции ошибки Хьюбера (Huber loss)**:

$$\text{huber}_\delta(z) = \begin{cases} \frac{1}{2}z^2, & |z| \le \delta, \\ \delta\left(|z| - \frac{1}{2}\delta\right), & |z| > \delta. \end{cases}$$

В окрестности нуля её график является параболой (см. рис. XX.9), а при больших значениях аргумента она линейная (что делает её «более устойчивой к выбросам», аналогично MAE). Отметим, что эта функция ошибки имеет уже гиперпараметр δ, выбор которого тоже должен быть обоснован (он как раз и задаёт границу перехода MSE-ошибки в MAE). Непараметрическая функция, которая похоже на функцию Хьюбера, но используется существенно реже –

$$\text{logcosh} = \log\left(\frac{\exp(z) + \exp(-z)}{2}\right).$$

Она всюду дважды дифференцируема, в отличие от функции Хьюбера, а её производная равна гиперболическому тангенсу.

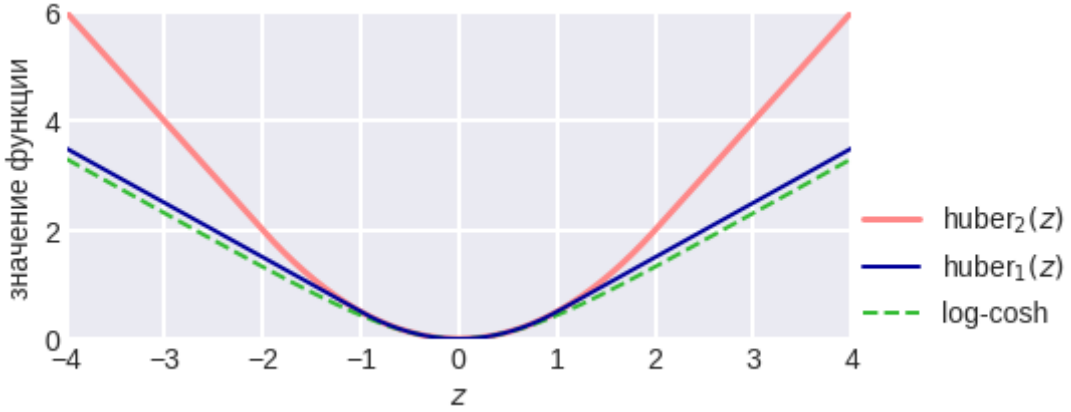

Рис XX.9. Функция ошибок Хьюбера и log-cosh.

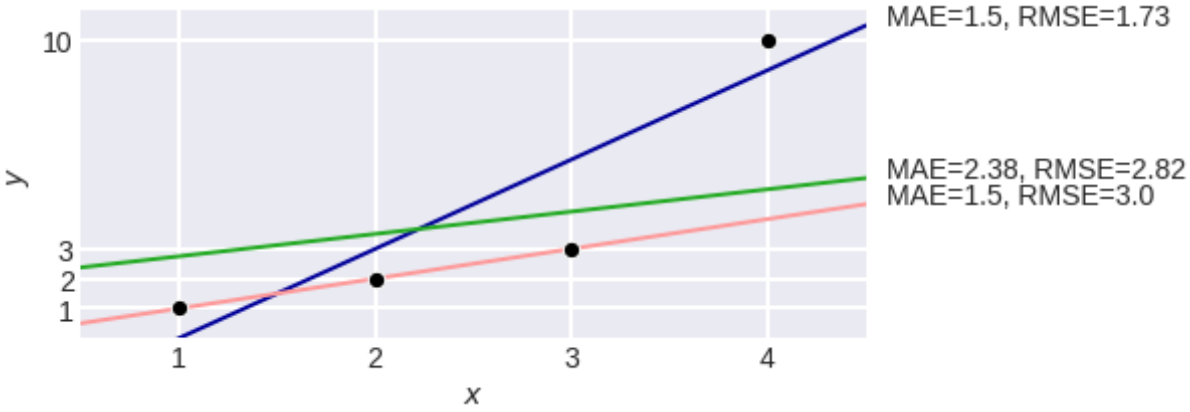

Рис. XX.10. Модельная задача регрессии.

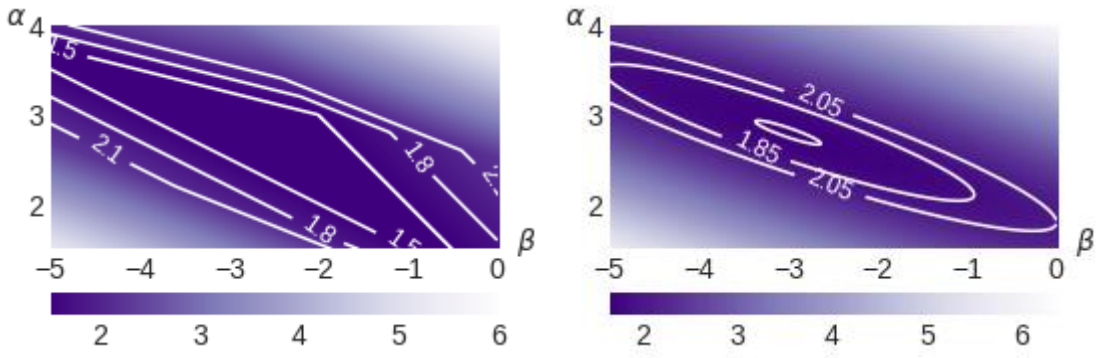

Рис. XX.11. Линии уровней функции ошибок MAE (слева) и RMSE (справа) в модельной задаче.

Для сравнения функций MAE и RMSE посмотрим ещё на следующую простую модельную задачу, см. рис. XX.10. На рис. XX.11 цветом показаны значения функций ошибок в зависимости от значения двух параметров прямой $y = \alpha x + \beta$. Заметим, что минимальное значение MAE равно 1.5 и оно достигается для всех пар параметров внутри треугольника с меткой «1.5» на рис. XX.11 (слева), минимальное значение RMSE достигается, естественно, в

Не противоречит ли это меньшей чувствительности MAE к выбросам?

одной точке (т.к. MSE – строго выпуклая функция). На рис. XX.10 видно, что минимальное значение MAE может быть у совершенно разных решений (при точной настройке на трёх объектах и при отклонении к выбросу).

Обобщения MAE и RMSE можно записать формулой

$$\left(\frac{1}{m} \sum_{i=1}^{m} w_i \, | \varphi(a_i) - \varphi(y_i) |^p \right)^{1/p}. \qquad \text{(XX.02)}$$

Наличие функции φ (как правило, используется логарифм) не должно смущать: преобразовав с помощью этой функции целевой вектор мы переходим к задаче с функцией ошибки без φ. Осталось не забыть после настройки модели и получения ответов, подвергнуть их обратному преобразованию:

$$\varphi^{-1}(a(x)).$$

Такие функции деформации φ часто используют, чтобы «привести регрессионные метки в более равномерную шкалу», см. главу «Визуализация» (например, чтобы ошибки на объектах с большими значениями меток не сильно портили решение). Самая популярная функция семейства (XX.02) – **Root Mean Squared Logarithmic Error (RMSLE[1]):**

$$\text{RMSLE} = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (\log(a_i + 1) - \log(y_i + 1))^2}.$$

Значения w_i в (XX.02), по сути, являются весами объектов, они неотрицательны, чаще всего их нормируют, чтобы в сумме они давали единицу:

$$w_1, w_2, \ldots, w_m \geq 0, \quad \sum_{i=1}^{m} w_i = 1.$$

Некоторые методы обучения позволяют **в явном виде задать веса объектов** (при настройке модели). В методах, основанных на сэмплировании (взятии подвыборок обучающей выборки), можно **проводить сэмплирование с вероятностями включения объектов в подвыборку, пропорциональными указанным весам**.

Способ сэмплирования – это тоже гиперпараметр настройки модели!

[1] Интересно, что в библиотеке numpy даже функция logp1 появилась из-за частой потребности аналитиков данных делать деформацию вида log(z+1), а также обратная к ней функция expm1.

Чуть сложнее с нетривиальными значениями p, но современные библиотеки оптимизации позволяют работать с любыми функционалами. Есть и «искусственные» способы решения, скажем, при $p=3$ можно сначала обучить алгоритм для функции ошибки при $p=2$, т.е. учитывая предыдущие замечания, модель настраивается на RMSE, быть может, с деформированным целевым признаком и весами объектов. После настройки можно в окрестности полученных параметров модели поискать меньшее значение исходной функции ошибки.

Рассмотрим, например, модельную задачу с целевыми метками 1, 2, 3, 6, 9. На рис. XX.12-13 показана ошибка разных константных алгоритмов для разных p. Оптимальные константные решения при $p=1$ и $p=0.5$ совпадают. При $1 \le p \le 2$ оптимальное константное решение смещается от медианы к среднему, а при увеличении p – к mid-range (среднему арифметическому максимума и минимума), см. рис. XX.14.

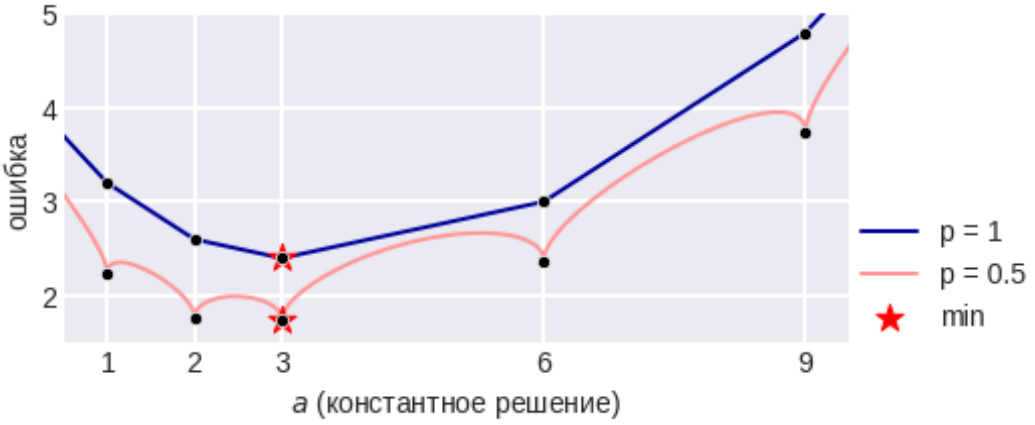

Рис. XX.12. Обобщённая ошибка константных решений для меток 1, 2, 3, 6, 9.

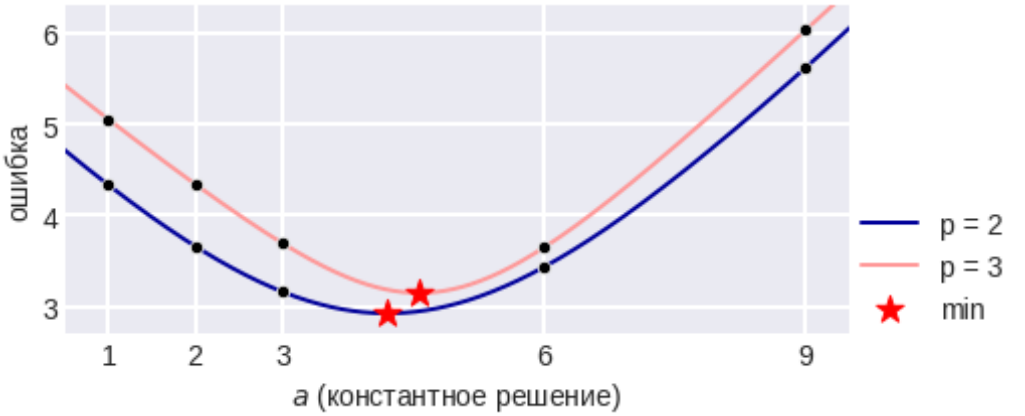

Рис. XX.13. Обобщённая ошибка константных решений для меток 1, 2, 3, 6, 9.

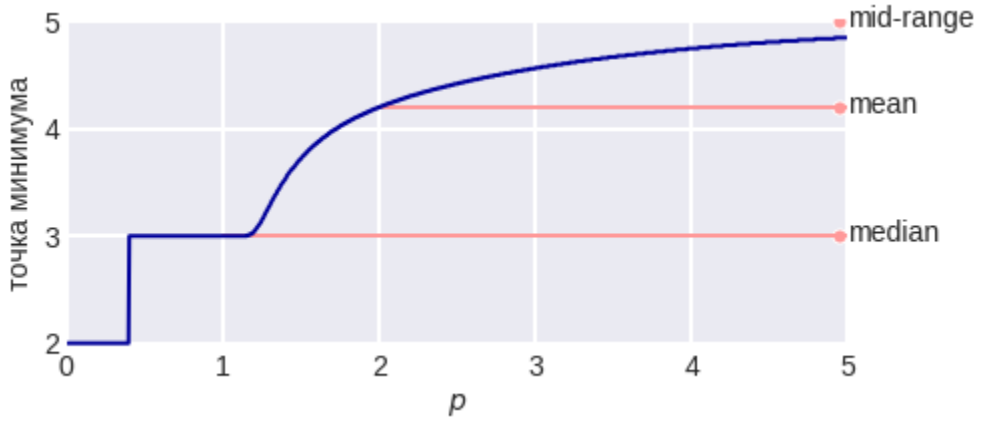

Рис. XX.14. Зависимость оптимального константного решения от степени.

Следующие функции пытаются измерить ошибку в процентах.

Средний процент отклонения (MAPE – Mean Absolute Percent Error):

$$\text{MAPE} = 100\% \cdot \frac{1}{m}\sum_{i=1}^{m}\frac{|y_i - a_i|}{|y_i|}.$$

Симметричный средний процент отклонения (SMAPE – Symmetric Mean Absolute Percentage Error):

$$\text{SMAPE} = \frac{2}{q}\sum_{i=1}^{m}\frac{|y_i - a_i|}{y_i + a_i} = 100\% \cdot \frac{1}{m}\sum_{i=1}^{m}\frac{|y_i - a_i|}{(y_i + a_i)/2}.$$

Эти функции ошибок выглядят очень логично: модуль разницы между ответом алгоритма и истинным значением мы на что-то нормируем: на модуль истинного значения или на среднее истинного и предсказанного. Будем считать, что значения целевого признака и ответы алгоритма положительные. При вычислении SMAPE можно считать, что они неотрицательные:

- если $y_i = a_i = 0$, тогда неопределённость вида 0/0 следует считать 0.
- при $a_i > y_i = 0$ или $y_i > a_i = 0$, получаем, что ошибка составляет 200%.

Вторая ситуация не кажется логичной и именно она вызывает проблемы в ситуациях, когда целевые значения часто равны нулю и мы используем SMAPE: при настройке алгоритма мы сосредотачиваемся на «угадывании нулей» (а не настраиваемся на ненулевые целевые значения).

Менеджеры часто просят посчитать ошибку в процентах, обосновывая это её интуитивностью. Почти все они впадают в ступор при вопросе, сколько процентов составляет 2 от 0?

Функции SMAPE и MAPE часто используют при прогнозировании временных рядов, особенно финансовых, где как раз разумно измерять ошибку в процентах. Например, рассмотри две ситуации: вместо цены за акцию 1 у.е. мы предсказали 2 и вместо 101 у.е. – 102. В первом случае мы ошиблись почти в два раза, а во втором «почти угадали», тем не менее MAE-ошибка в обоих случаях равна 1, а вот (S)MAPE сильно различается.

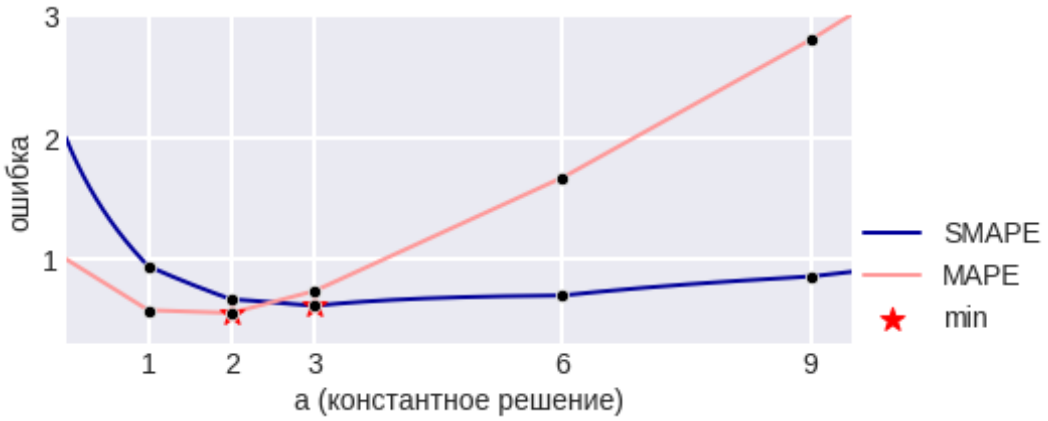

Рис. XX.15. SMAPE и MAPE-ошибки константных решений для меток 1, 2, 3, 6, 9.

На рис. XX.15 показано качество константных решений для модельной выборки с целевыми значениями 1, 2, 3, 6, 9. Но такая ситуация не очень практическая: целевые значения редко так сильно отличаются друг от друга (в 9 раз!), поэтому на рис. XX.16 показана более реалистичная ситуация.

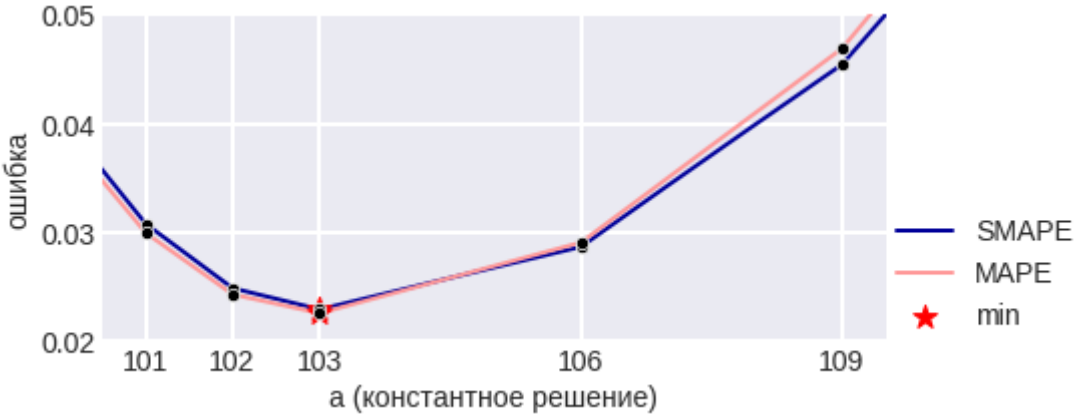

Рис. XX.16. SMAPE и MAPE-ошибки константных решений для меток 101,102, 103, 106, 109.

Заметим, что на рис. XX.16 оптимальное решение – медиана. Действительно, на практике при работе с (S)MAPE медиана часто полезна, например, для усреднения алгоритмов в ансамбле. Отметим также, что хотя MAPE и SMAPE похожи, с точки зрения практической минимизации между ними большое отличие, первую функцию можно переписать в таком виде:

$$\text{MAPE} = \frac{1}{m}\sum_{i=1}^{m} w_i \,|\, y_i - a_i \,|\text{, где } w_i = \frac{1}{|\, y_i \,|},$$

здесь существенно то, что веса w_i определяются постановкой задачи и не зависят от ответов алгоритма (их можно вычислить по обучающей выборке и производить обучение, учитывая их значения), т.е. MAPE – это просто весовой MAE.

Есть и другие способы перевода модуля отклонения в проценты, например

$$\text{PMAD} = \frac{\sum_{i=1}^{m} |\, y_i - a_i \,|}{\sum_{i=1}^{m} |\, y_i \,|}$$

но эта функция отличается от MAE константным множителем (и с точки зрения минимизации они эквивалентны).

Есть целый класс функций ошибок, которые основаны **на сравнении ошибки с некоторым бенчмарком** (уже существующим надёжным алгоритмом решения задачи). Пусть a_i' – метки, полученные бенчмарком.

MRAE – Mean Relative Absolute Error:

$$\text{MRAE} = \frac{1}{m}\sum_{i=1}^{m} \frac{|\, y_i - a_i \,|}{|\, y_i - a_i' \,|}.$$

Это среднее отношений ошибок на разных объектах.

Можно также поделить суммарные ошибки:

$$\text{REL\_MAE} = \frac{\sum_{i=1}^{m} |\, y_i - a_i \,|}{\sum_{i=1}^{m} |\, y_i - a_i' \,|},$$

или посчитать, в каком проценте случаев алгоритм лучше бенчмарка – это делает функционал качества **Percent Better**:

$$\text{PB(MAE)} = \frac{1}{m}\sum_{i=1}^{m} I[\,|\, y_i - a_i \,| < |\, y_i - a_i' \,|\,].$$

В качестве бенчмарка можно брать

- простой константный алгоритм (часто используют нулевой, сравните MAPE и MRAE),
- предыдущую версию Вашего алгоритма (чтобы оценить, насколько улучшилось решения),
- естественный алгоритм (в задачах прогнозирования рядов самый естественный и простой прогноз: значение функции в следующий момент времени такое же, как в предыдущий).

Mean Absolute Scaled Error иллюстрирует последнее замечание:

$$\text{MASE} = \frac{1}{\frac{m}{m-1}\sum_{i=2}^{m}|y_i - y_{i-1}|}\sum_{t=1}^{m}|y_i - a_i|$$

Здесь везде модуль разности между значением, полученным алгоритмом, и истинным можно заменить на модуль в некоторой степени. Ещё один функционал качества (уже не функция ошибки): процент случаев, когда ответ алгоритма верен с некоторой заранее заданной точностью

$$\text{eB} = \frac{1}{m}\sum_{i=1}^{m} I[|y_i - a_i| \le \varepsilon].$$

Рис. XX.17. eB-качество константных решений для меток 1,2, 3, 6, 9.

Оптимальное константное значение для этого функционала мы фактически научились строить в главе «Средние». График этого функционала в зависимости от ответа константного алгоритма показан на рис. XX.17, по сути это график парзеновской оценки плотности выборки (с точностью до

нормировки). Оптимальное константное решение соответствует моде построенной оценки плотности.

На рис. XX.18 показаны значения eB-качества в модельной задаче рис. XX.10 в зависимости от параметров линейной регрессии. Естественно, здесь уже качество не выпукло по параметрам.

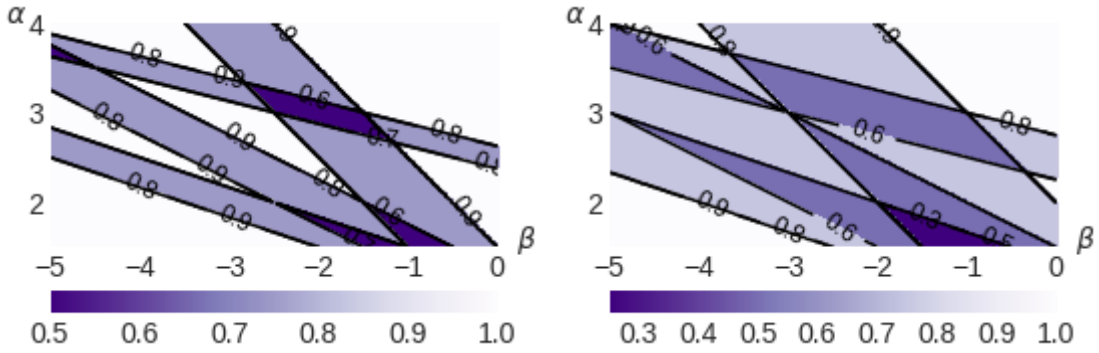

Рис. XX.18. Линии уровней функции eB-качества для $\varepsilon = 0.5$ (слева) и $\varepsilon = 1$ (справа) в модельной задаче.

Стоит обратить внимание, что не всегда функции ошибки симметричные. На практике часто ошибки «в разные стороны» (завышение и занижение) неравнозначны. Например, при прогнозировании спроса на товар: если наш алгоритм занижает число продаж, то мы рискуем, что товар быстро раскупят и будут недовольные клиенты, оставшиеся без товара. Если же завышает, то мы не сможем продать весь товар, он останется на складе (в дальнейшем его придётся продавать по сниженным ценам или возвращать поставщику). С точки зрения денежных потерь, «завысить прогноз на 10» и «занизить на 10» имеют разную стоимость. В таких случаях используют несимметичные функции ошибок, см. рис. XX.19:

$$\frac{1}{m}\sum_{i=1}^{m}\begin{cases} g(|\,y_i - a_i\,|), & y_i < a_i, \\ h(|\,y_i - a_i\,|), & y_i \geq a_i. \end{cases}$$

На рис. XX.20 показана ошибка константного алгоритма для функции

$$\frac{1}{m}\sum_{i=1}^{m}\begin{cases} k_1(y_i - a_i), & a_i < y_i, \\ k_2(a_i - y_i), & a_i \geq y_i, \end{cases}$$

При $k_1 = k_2$ оптимальной константой была бы медиана. При $k_1 < k_2$ мы больше штрафуем завышение и оптимальная константа смещается влево, при $k_1 > k_2$ больше штрафуется занижение и оптимальная константа смещается вправо, но она всегда остаётся представителем выборки.

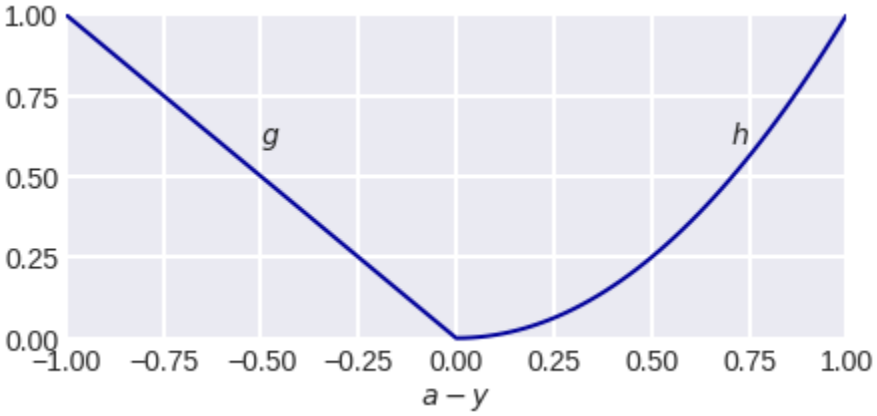

Рис. XX.19. Пример несимметричной функции ошибки.

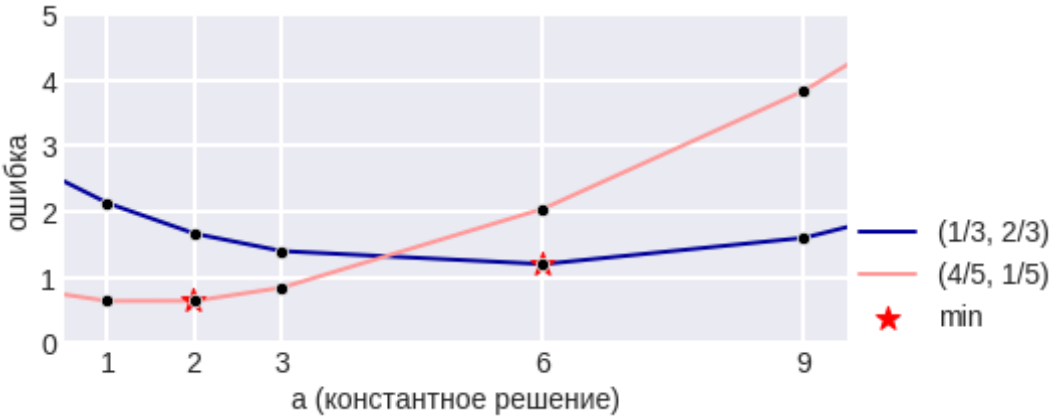

Рис. XX.20. Несимметричные ошибки константных решений для меток 1, 2, 3, 6, 9.

На самом деле, рассмотренная нами функция ошибки – т.н. ошибка **квантильной регрессии (Quantile Regression)**, её минимизирует τ - квантиль, $\tau = k_2 / (k_1 + k_2)$.

Ошибки в регрессии: приложения

1. На практике сложно адекватно расписать штраф за недопрогноз и перепрогноз в несимметричных функциях ошибки. Например, при выполнении одного индустриального проекта на вопрос «что хуже и во сколько раз: превысить в итоговом химическом составе допустимую долю серы или принизить» сталевар однозначно ответил «не допустимы оба варианта» (хотя по статистике они случались на практике). Если рассмотреть бытовые ситуации: заливаем в бензобак количество бензина a при необходимом на дорогу y, то понимаем, что тут при $a < y$ штраф должен быть бесконечный, т.к. мы просто не доедем до цели. Или прийти к поезду, уходящему в момент

времени y, в момент a – аналогично, при $a > y$ мы опаздываем и штраф бесконечный. Когда же приходим заранее, не понятно как должна вести себя функция ошибки: до некоторого момента, возможно, линейно, но приходить за 2 часа нам не хочется (и тут тоже есть соблазн сделать бесконечный штраф). Поэтому **не надейтесь, что функцию ошибки придумает эксперт!** Мы сами не можем её придумать для знакомых нам ситуаций.

2. **Функции ошибки могут пригодиться не только для непосредственного измерения ошибки**, например, с их помощью можно также генерировать признаки в некоторых задачах машинного обучения. Например, в задаче классификации сигналов сигнал часто описывают с помощью признаков. В такое описание могут включать статистические характеристики сигнала (среднее, дисперсию и т.п.), описание некоторой модели, которая представляет сигнал (например, коэффициенты в разложении Фурье) и т.д. Можно также оценивать, насколько сигнал «предсказуем»: строить модель сигнала (возможно, по его части) и сравнивать прогноз модели с истиной. Ошибка прогноза может оказаться неплохим признаком.

Старайтесь стандартным средствам находить нестандартные применения.

3. В соревновании «Rossmann Store Sales» на платформе Kaggle[1] была использована функция ошибки **Root Mean Square Percentage Error (RMSPE)**:

$$\sqrt{\frac{1}{|\{i \mid y_i > 0\}|}\sum_{i:\, y_i>0}\left(\frac{a_i - y_i}{y_i}\right)^2}$$

(тут вычисление идёт по всем неотрицательным целевым значениям), один из участников обосновал **оптимальную деформацию целевого признака, которая позволяла решить задачу одной из стандартных моделей алгоритмов машинного обучения**. Ищем деформацию F после применения которой

$$\frac{a-y}{y} \approx F(a) - F(y),$$

т.е. функция ошибки превращается в RMSE:

$$\sqrt{\frac{1}{|\{i \mid y_i > 0\}|}\sum_{i:\, y_i>0}(F(a_i) - F(y_i))^2}\ .$$

[1] https://www.kaggle.com/c/rossmann-store-sales

Как отмечалось выше на минимизацию (R)MSE заточено большинство стандартных алгоритмов регрессии. Пусть $a = y + \delta$, где δ мало, тогда

$$\frac{\delta}{y} \approx F(y+\delta) - F(y) = F'\delta + o(\delta),$$

решим уравнение

$$\frac{\delta}{y} = F'\delta,$$

$$\frac{1}{y} = \frac{\partial F}{\partial y},$$

Знание методов решения дифференциальных уравнение вдруг становится полезным при выводе оптимальной деформации.

получили решение в виде $F(y) = \ln|y| + C$. Интересно, что в рассматриваемой задаче после деформации $F(y) = \ln(y+1)$ ненулевые целевые значения стали почти нормально распределены, см. рис. XX.21.

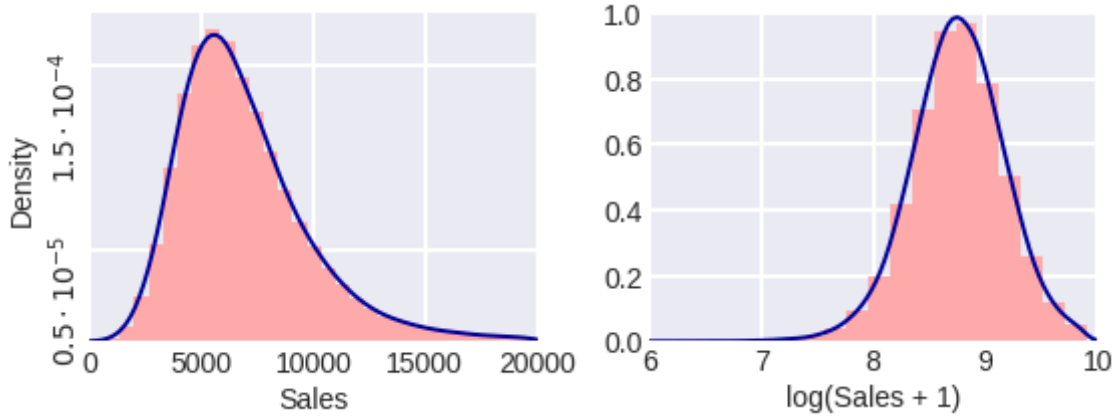

Рис. XX.21. Распределение ненулевых значений целевого признака (Sales) до (слева) и после (справа) деформации.

Вопросы и задачи

1. Верно ли, что любая функция ошибки соответствует некоторому распределению вероятностей (как MSE – нормальному, а MAE – распределению Лапласа)?

2. Приведите пример задачи регрессии, в которой решение полученное оптимизацией MAE более чувствительно к выбросам, чем решение полученное оптимизацией MSE.

3. Что лучше оптимизировать методом градиентного спуска: MSE или RMSE?

4. Можно ли использовать линейное программирование при настройке линейной регрессии с обобщённой функцией ошибки (XX.02) при $p=1$? А при $p=+\infty$?

5. Рассмотрите такую функцию ошибки[1]:

$$\frac{1}{m}\sum_{i=1}^{m}\log(1+|a_i-y_i|^2).$$

Соответствует ли ей какое-то распределение? В чём может быть её преимущество по сравнению с рассмотренными ранее?

6. Рассмотрим задачу «взлома лидерборда». Вы участвуете в соревновании по машинному обучению, можете отправлять решение – предсказания целевых значений на фиксированной выборке: $a_1,\ldots,a_m$, в ответ получаете значение MAE / MSE, вычисленное по истинным значениям $y_1,\ldots,y_m$. За сколько отправок можно гарантированно узнать все целевые значения? Например, если

$$\text{MAE}((0,0),(y_1,y_2))=(|y_1|+|y_2|)/2=1,$$

$$\text{MAE}((1,0),(y_1,y_2))=(|1-y_1|+|y_2|)/2=1.5,$$

то ещё нет однозначной уверенности в целевых значениях, допустимы значения $y_1=-2,\ y_2=0$ и $y_1=0,\ y_2=\pm 2$. Если же, в дополнение,

$$\text{MAE}((1,3),(y_1,y_2))=(|1-y_1|+|3-y_2|)/2=2,$$

то целевые значения определены однозначно: $y_1=-1,\ y_2=1$.

[1] См. например Saleh R. A., Saleh A. K. Statistical properties of the log-cosh loss function used in machine learning //arXiv preprint arXiv:2208.04564. – 2022.

Итоги

- Функции ошибки могут иметь вероятностное обоснование через метод максимального правдоподобия. Важно распределение невязок ($a - y$).
- Функции ошибки в задачах регрессии[1]: MSE, MAE и их обобщения и модификации, процентные (SMAPE, MAPE), сравнение с бенчмарком (MRAE, PB, MASE), несимметричные, пороговые (eB).
- Есть нетрадиционные применения функций ошибок.

[1] См. также

Стрижов В.В. Функция ошибки в задачах восстановления регрессии // Заводская лаборатория, 2013, 79(5): 65-73.

Wang Q. et al. A comprehensive survey of loss functions in machine learning //Annals of Data Science. – 2020. – С. 1-26.

Код к главе

Ниже представлены функции, реализованные в библиотеке scikit-learn, а также их прямые реализации (используя функции numpy). Из функций ошибок, которые мы в явном виде не упомянули, там реализована **Median Absolute Error**:

$$\text{MedAE} = \text{median}(|a_1 - y_1|,\ldots,|a_m - y_m|).$$

```
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_squared_log_error
from sklearn.metrics import median_absolute_error
from sklearn.metrics import explained_variance_score

# R^2
print (r2_score(y, a),
       1 - np.mean((y - a) ** 2) / np.mean((y - np.mean(y)) ** 2))
# MAE
print (mean_absolute_error(y, a),
       np.mean(np.abs(y - a)))
# MSE
print (mean_squared_error(y, a),
       np.mean((y - a) ** 2))
# MSLp1E
print (mean_squared_log_error(y, a),
       np.mean((np.log1p(y) - np.log1p(a)) ** 2))
# MedAE
print (median_absolute_error(y, a),
       np.median(np.abs(y - a)))
```

Спасибо за внимание к книге!
Замечания по содержанию, замеченные ошибки и неточности можно написать в телеграм-чате
https://t.me/Dyakonovsbook